Homework 2 Group Part Report

Team: Miguel Fuentes, Dan Cortes, Vineeth Gutta, Praveena Adavikolanu

Naive Bayes

Data Import

In [1]:

```
import pandas as pd # this library is used for the dataframe data structure

train1 = pd.read_pickle('data\\dataset 1\\train.pkl')
test1 = pd.read_pickle('data\\dataset 1\\test.pkl')

train2 = pd.read_pickle('data\\dataset 2\\train.pkl')
test2 = pd.read_pickle('data\\dataset 2\\test.pkl')

train3 = pd.read_pickle('data\\dataset 3\\train.pkl')
test3 = pd.read_pickle('data\\dataset 3\\test.pkl')
```

Model Results

In [4]:

```
import NaiveBayes # this holds our implementation of naive bayes

model1 = NaiveBayes.NaiveBayes()
model1.train(train1, ['spam', 'ham'])
print(f'Accuracy on data set 1: {NaiveBayes.accuracy(model1, test1)}')

model2 = NaiveBayes.NaiveBayes()
model2.train(train2, ['spam', 'ham'])
print(f'Accuracy on data set 2: {NaiveBayes.accuracy(model2, test2)}')

model3 = NaiveBayes.NaiveBayes()
model3.train(train1, ['spam', 'ham'])
print(f'Accuracy on data set 3: {NaiveBayes.accuracy(model3, test3)}')
```

```
Accuracy on data set 1: 0.9602510460251046
Accuracy on data set 2: 0.956140350877193
Accuracy on data set 3: 0.9300184162062615
```

Logistic Regression

Data Import and Formatting

The following block of code is used to process the data into the form which is appropriate for logistic regression

In [10]:

```
from collections import namedtuple

import spam_ham_util # this holds a data formatting utility

DataSet = namedtuple('DataSet',['X_train', 'X_test', 'Y_train', 'Y_test'])

dataset1 = DataSet(*spam_ham_util.df_to_numeric(train1, test1))
dataset2 = DataSet(*spam_ham_util.df_to_numeric(train2, test2))
dataset3 = DataSet(*spam_ham_util.df_to_numeric(train3, test3))
```

Hyperparamater Tuning

In the following block we test various values for the regulerization constant and see which one results in the best accuracy on a 70/30 split of the training data

In [13]:

```
# This accuracy calculator was not written by us
from sklearn.metrics import accuracy_score
from sklearn.model_selection import train_test_split

import log_regression # this holds our implementation of logistic regression

ITERS, LEARNING_RATE = 100000, 0.01

for l2_reg in [0, 0.15, 0.3, 0.45, 0.6, 0.75]:
    accs = []
    for dataset in [dataset1, dataset2, dataset3]:
        X_train, X_test, Y_train, Y_test = train_test_split(dataset.X_train, dataset.Y_train, test_
size=0.30, random_state=17)

        model = log_regression.log_regression(ITERS, l2_reg, LEARNING_RATE)
        model.train(X_train, Y_train)

        accs.append(accuracy_score(model.predict(X_test), Y_test))
    print(f'Average accuracy with l2_reg = {l2_reg} is: {sum(accs)/len(accs)}')
```

```
Average accuracy with l2_reg = 0 is: 0.9254289873829026
Average accuracy with l2_reg = 0.15 is: 0.9351634220573851
Average accuracy with l2_reg = 0.3 is: 0.9334917711103717
Average accuracy with l2_reg = 0.45 is: 0.9434449955563501
Average accuracy with l2_reg = 0.6 is: 0.9420299785016707
Average accuracy with l2_reg = 0.75 is: 0.9365073848443676
```

Test Set Accuracy

Now that the regularization constant has been chosen to be 0.45 we will train on the entire training set and report accuracy on each of the test sets

In [14]:

```
best_reg_val = 0.45

for index, dataset in enumerate([dataset1, dataset2, dataset3]):
    model = log_regression.log_regression(ITERS, best_reg_val, LEARNING_RATE)
    model.train(dataset.X_train, dataset.Y_train)

    print(f'Accuracy of dataset{index + 1} is : {accuracy_score(model.predict(dataset.X_test),
dataset.Y_test)}')
```

```
Accuracy of dataset1 is : 0.9058577405857741
Accuracy of dataset2 is : 0.9057017543859649
Accuracy of dataset3 is : 0.9631675874769797
```

Perceptron Algorithm

This code will report the optimal hyperparamaters and the accuracy when trained on all the training data for every dataset individually

In [1]:

```
import perceptron # this is our implementation of the perceptron algorithm

print("DATASET 1---------------------------------")
perceptron.main('data\\dataset 1\\train', 'data\\dataset 1\\test')
print("\nDATASET 2---------------------------------")
perceptron.main('data\\dataset 2\\train', 'data\\dataset 2\\test')
print("\nDATASET 3---------------------------------")
perceptron.main('data\\dataset 3\\train', 'data\\dataset 3\\test')
```

```
DATASET 1---------------------------------

Data initialized:
Started training...

Best results from hyperparameter tuning:
Best Acc:  100.0
best Epoch:  75
Best lr:  0.01

RESULTS FROM ENTIRE DATASET:

Learning rate: 0.0100
Number of epochs: 75
Emails classified correctly: 443/478
Accuracy: 92.6778%

DATASET 2---------------------------------

Data initialized:
Started training...

Best results from hyperparameter tuning:
Best Acc:  100.0
best Epoch:  75
Best lr:  0.01

RESULTS FROM ENTIRE DATASET:

Learning rate: 0.0100
Number of epochs: 75
Emails classified correctly: 414/456
Accuracy: 90.7895%

DATASET 3---------------------------------

Data initialized:
Started training...

Best results from hyperparameter tuning:
Best Acc:  100.0
best Epoch:  75
Best lr:  0.01

RESULTS FROM ENTIRE DATASET:

Learning rate: 0.0100
Number of epochs: 75
Emails classified correctly: 501/543
Accuracy: 92.2652%
```